Naam: Rune Heus Rolnummer: 0600234 E-mail: rune.heus@vub.be Academiejaar: 2023-2024

VRIJE UNIVERSITEIT BRUSSEL

Specificatie: Modelspoor

1. Introductie

Dit document beschrijft het ontwerp van de implementatie voor het ontwikkelen van een controlesysteem voor een modelspoor in de programmeertaal Racket voor het opleidingsonderdeel “Programmeerproject 2 fase 3”.

2. Functionaliteit

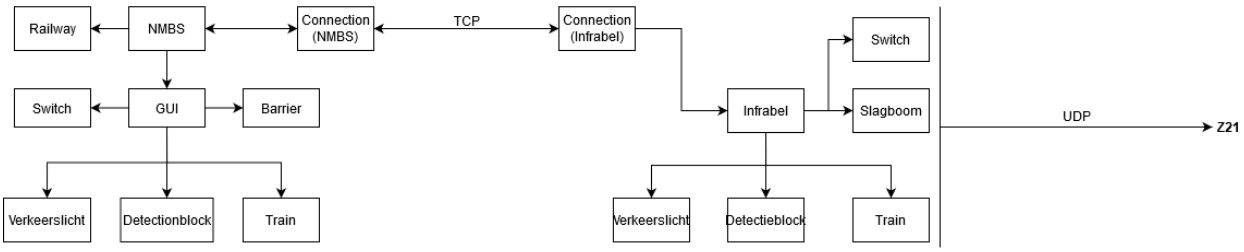
De implementatie van het project dat beschreven wordt in dit document, bestaat uit drie grote componenten: Een Command Station (Z21), Infrabel en NMBS (+ GUI). De Z21 is de hardware dat de verschillende elementen van het modelspoor zal aansturen. Infrabel ligt tussen de Z21 en NMBS en zal communiceren tussen de software en hardware (modelspoor). NMBS zal communiceren met Infrabel en staat in voor de functionaliteit en de GUI dat geen logisch onderdeel is.

Uitgebreide vereisten

Als uitgebreide vereisten heb ik gekozen voor ’**Scenarios inladen of opslaan met de simulator’.** Hiervoor zal er een file (genaamd saved_scenario.txt) gemaakt worden, telkens het programma afgesloten wordt. In deze file zullen alle ADT’s die verantwoordelijk zijn voor een scenario, opgeslagen worden in een formaat dat zowel de software als mens kan lezen en begrijpen. De informatie die wordt opgeslagen zijn:

- De toegevoegde treinen, hun positie, snelheid en of ze een bestemming hadden of niet
- De lichten, hun id en hun status (groen of rood)
- De barrières, hun id en hun status (open of dicht)
- De switches, hun id, de inkomende rail, de uitgaande rail en de uitgaande rail die niet actief is
- De detectieblokken, hun id, de train die aanwezig is, wat de volgende trainen zijn die er eventueel op kunnen zitten en of er überhaupt een train op zit.

3. Software Architectuur



We zien hier in de figuur dat we 2 (eigenlijk 3, als we de interface mee tellen) aparte delen in de architectuur hebben. We hebben het NMBS deel en het Infrabel deel, deze 2 delen zijn afgezonderd door een TCP connectie. Deze TCP connectie zal zorgen voor het versturen en ontvangen van berichten die doorgestuurd kunnen worden door de ADT's van zowel NMBS als Infrabel.

De werking van deze connectie loopt als volgt: We zullen eerst aan de kant van Infrabel, een server starten die zal wachten voor een binnenkomende connectie (in dit geval van NMBS). Als deze connectie vastgelegd is, kunnen de 2 partijen met elkaar communiceren.

De communicatie zal gebeuren als een ADT van een van de partijen een bericht wil sturen naar de andere partij, dit kan gaan over het toevoegen van treinen op de simulator of hardware, of het starten of stoppen van een trein, het veranderen van switches, open of sluiten van barrières, … . Wanneer dit gebeurt, zal dit bericht doorgegeven worden naar het connectie ADT van de huidige partij en deze connectie zal dit bericht doorsturen naar de connectie van de andere partij (Door het bericht in de output port te zetten en deze te flushen). Vervolgens zal elke connectie om de 10 ms kijken of er een bericht is binnengekomen via hun input port. Als dit zo is zal dit bericht onderzocht worden, gekeken worden naar welke ADT dit bericht bedoeld is en vervolgens dit bericht met de meegegeven waardes doorgeven naar de juiste procedure in het juiste ADT (Vice versa).

Door gebruik te maken van TCP hebben geen enkele partij rechtstreeks procedure, waardes of ADT's nodig van de andere partij en kan dit apart allemaal geregeld worden door de connecties.

NMBS

Als we gaan kijken naar de NMBS kant zien we dat we verschillende ADT's hebben die hier bij horen:

- Railway
- Gui
- Light
- Barrier
- Detectionblock
- Train
- Switch

Als we naar de afbeelding gaan kijken zien we dat de GUI vooral verantwoordelijk is voor het bijhouden van de instanties van de ADT's. Ik heb dit zo gedaan dat alle instanties centraal op 1 plaats terug te vinden zijn. De Gui is in dit geval ook verantwoordelijk dat de gebruiker het treinspoor kan besturen (alles over het besturen van het treinspoor is terug te vinden in de handleiding). Het NMBS ADT is het centraal ADT van heel de NMBS kant. Het NMBS zal de instantie van de Gui bijhouden, de connectie en

het railway ADT. Het NMBS ADT is ook verantwoordelijk om de connectie instantie door te geven aan de rest van de ADT's zodat deze er ook gebruik van kunnen maken.

De werking van alle ADT's van NMBS kan je terug vinden onder sectie 4.1.

Infrabel

De onderverdeling van de ADT's bij Infrabel zijn gelijkaardig aan die van NMBS. Het Infrabel ADT gaat in dit geval verantwoordelijk zijn voor de instanties van alle andere ADT's die nodig zijn voor het spoor.

Je kunt in de afbeelding ook zien dat alle ADT's een connectie zullen hebbe via UDP naar de interface, dus deze hoeven niet apart doorgestuurd te worden naar een connectie of Infrabel om berichten te versturen.

De connectie voor Infrabel is gelijkaardig aan die van NMBS, het verschil is dat de connectie van Infrabel als server zal werken en die van NMBS als een client (je kunt dus zowel van server naar client als van client naar server).

4. ADT's

Alleen de belangrijke en relevante procedure zijn terug te vinden met hun signatuur.

1) NMBS

Voor elke aanwezig ADT in NMBS zal er een bericht via de NMBS connectie verstuurd worden naar Infrabel, zodat hier ook Infrabel instanties van kunnen gemaakt worden.

a) NMBS

Hierin zal de connectie, gui en de railway instanties gemaakt worden. Het NMBS ADT is verantwoordelijk voor het opzetten en klaarmaken (Het setuppen van de ontvangen/aanwezige switches, detectieblokken, …) van de GUI en het spoor langs de kant van NMBS. Telkens de interface nieuwe informatie voor ons heeft (zoals bijvoorbeeld, als een detectieblock een nieuwe trein op zich heeft), zal NMBS hier rekening met houden en er voor zorgen dat dit allemaal up-to-date blijft. Het ADT is ook verantwoordelijk voor het checken van trein condities (heeft een trein zijn bestemming bereikt? Moeten we aan de detectieblok laten weten welke exacte trein er zich bevindt?).

Naam	Signatuur	Beschrijving
make-NMBS	**NMBS**	**Constructor**
start	(Ø -> Ø)	Dit zal de NMBS loop starten.
setup-switches	(List -> Ø)	Zal de aanwezige switches van het spoor toevoegen.
setup-detectionblocks	(List -> Ø)	Zal de aanwezige detectieblokken van het spoor toevoegen.
start-route	List -> Ø	Zal de trein die meegegeven is in de lijst laten sturen naar het pad die ook in de lijst zit.

detectionblock-loop	List -> Ø	Gaat kijken welke detectieblokken er geupdate moeten worden, (of er wel of niet zich een trein bevindt).

b) GUI

Dit representeert het pop-up scherm bovenop het modelspoor waar de gebruiker alle functionaliteiten van het spoor kan aansturen zoals: Trein toevoegen, trein van rijrichting veranderen, trein versnellen of vertragen, slagbomen besturen, etc. Hierin zullen ook alle instanties van de nodige ADT voor het spoor opgeslagen worden. Dit speelt dus een belangrijke rol. Hiervoor is er gebruik gemaakt van de Rackter/gui library.

Naam	Signatuur	Beschrijving
make-gui	**(Connection -> Gui)**	**Constructor**
add-train!	(Ø -> Ø)	Zal een trein toevoegen aan de train en speed tab.
remove-train!	(Symbol -> Ø)	Zal de trein met meegegeven id verwijderen.
add-switch!	(Symbol -> Ø)	Zal een switch toevoegen met meegegeven id.
add-light!	(Symbol, Symbol -> Ø)	Zal een licht toeven met het meegegeven id en de status van dit licht gelijk zetten aan het meegegeven code symbool.
set-light-value!	(Light, Symbol -> Ø)	Zet de status van het meegegeven licht gelijk aan de meegegen code.
add-detectionblock!	(Symbol -> Ø)	Zal een detectieblock toevoegen met het meegegeven id.
change-detection-block!	(Symbol, Boolean -> Ø)	Zal de status van de detectieblock van het meegegeven id veranderen naar de meegegeven waarden.
add-barrier!	(Symbol -> Ø)	Zal een slagboom toevoegen met het meegegeven id.
inform-detectionblocks	(Train, Symbol) -> Ø	Dit zal alle detectieblokken die de meegegeven trein als next-train bevatten laten weten dat ze deze niet meer moeten onthouden, dit zal gebeuren voor alle buiten de meegegeven detectieblok id.
change-train-position!	Symbol -> Ø	Past de positie van de trein met meegegeven id aan.

c) Train

Dit is de trein ADT langs de NMBS kant. Hierin wordt het ID, de positie en de snelheid bijgehouden. Als er iets moet veranderen aan de trein op het spoor (aan de Infrabel kant), zal deze trein via het NMBS connectie ADT een bericht sturen naar de connectie van Infrabel, zodat dit aangepast kan worden in de interface.

Naam	Signatuur	Beschrijving
make-nmbs-train	**(Symbol (id) -> Train)**	**Constructor**
change-speed!	(Number -> Ø)	Zal een bericht toevoegen aan de messages voor infrabel om te vragen voor de snelheid van de trein aan te passen.
get-id	(Ø -> Symbol)	Geeft het id terug van de trein.
get-speed	Number	Geeft de snelheid terug.
set-pos!	Symbol -> Ø	Zet de positie van de trein gelijk aan het meegegeven symbool.
get-pos	Symbol	Geeft de positie van de trein terug.
get-id	Symbol	Geeft het id terug van de trein.
get-destination	Symbol	Geeft het bestemming van de trein terug.
set-destination!	Symbol -> Ø	Zet de eind bestemming gelijk aan het meegegeven symbool.

d) Detectieblok

Hierin zullen het ID, de aanwezige trein (in ADT vorm), de lijst van potentiële volgende treinen en of deze überhaupt een trein bevatten of niet, in worden opgeslagen. Om te weten welke trein er zich bevindt op een detectieblok, kun je dit zien in de “detectionblocks” tab in de GUI.

Naam	Signatuur	Beschrijving
make-nmbs-detectionblock	**(Symbol (id), Boolean (occupied?) -> Detectionblock)**	**Constructor**
occupied?	(Ø -> Boolean)	Zal terug geven of er een trein aanwezig is op het spoor of niet.
set-occupied!	(Boolean -> Ø)	Zal een bericht sturen naar infrabel om de status van de detectieblock aan te passen.
get-id	(Ø -> Symbol)	Geeft het id terug van de detectieblok.
add-train!	Train -> Ø	Zet de momentele actieve trein in de detectieblok.
remove-train!	Ø -> Ø	Verwijderd de actieve trein van de detectieblock.
train	Train	Geeft de momentele actieve trein terug.
next-trains	List	Geeft de lijst terug van de volgende treinen.
add-next!	Train -> Ø	Voegt de meegegeven trein toe aan de lijst van de volgende treinen.
remove-next-train!	Train -> Ø	Verwijderd de meegegeven trein uit de lijst van de volgende treinen.
set-next-trains-list	List -> Ø	Zal de lijst van volgende potentiële treinen aanpassen naar de meegegeven lijst.

e) Barrier

Hierin zullen het ID en of de slagboom open of dicht is in worden opgeslagen. Deze zal ook via het connectie ADT van NMBS een bericht sturen naar Infrabel, als de slagboom open of dicht moet.

Naam	Signatuur	Beschrijving
make-nmbs-barrier	**(Symbol (id), Boolean (closed?)-> Slagboom)**	**Constructor**
get-id	(Ø -> Symbol)	Geeft het id terug van de slagboom.
set-status!	(Boolean -> Ø)	Zal via Infrabel de status veranderen naar de meegegeven waarden.
closed?	Boolean	Geeft terug of de barriere gesloten is of niet.

f) Light

Voor het doorsturen van commando's dat te maken heeft met de verkeerslichten op het spoor. Hierin zullen ook het ID en de code (groen of rood) van het licht bewaart worden.

Naam	Signatuur	Beschrijving
make-nmbs-light	**(Symbol (id), Symbol (licht status) -> Verkeerslicht)**	**Constructor**
code	Symbol	Geeft de code van het licht terug (groen of rood)

g) Switch

Dit is het adt dat te maken heeft met het doorsturen van berichten naar infrabel dat te maken heeft met de switch op het spoor. Het ID, de binnenkomende rail, buitengaande rail en niet actieve buitengaande rail, zullen hier opgeslagen worden. Wanneer de status veranderd wordt van een switch zal deze de buitengaande en de niet actieve buitengaande rail van waarden switchen en dit via de connectie doorgeven aan Infrabel.

Naam	Signatuur	Beschrijving
make-nmbs-switch	**(Symbol (id) -> Switch)**	**Constructor**
get-id	(Ø -> Symbol)	Geeft het id terug van de switch.
change-status!	(Number -> Ø)	Zal een bericht sturen van naar infrabel voor het aanpassen van de status van de switch met de meegegeven waarden.
get-value	Boolean	Dit kun je gebruiken om te weten of de switch veranderd is van zijn originele waarden.

h) Connection

Dit is het connection adt dat voor het updaten van nmbs zal zorgen door de berichten voor NMBS in te lezen en af te handelen. Dit ADT zal als client werken en een connectie aanvragen aan de server van Infrabel. Als er berichten worden ontvangen via de input port zal de connectie deze behandelen naar het juiste ADT.

Naam	Signatuur	Beschrijving
make-nmbs-connection	**(NMBS -> Connection)**	**Constructor**
update-nmbs	(Ø -> Ø)	Zal de berichten voor NMBS afhandelen naar de NMBS componenten.
in	Port	Geeft de input port terug.
out	Port	Geeft de output port terug.
send-message	Lijst	Zal het meegegeven bericht via de output port doorsturen naar de server van Infrabel.

i) Railway

Dit de representatie van het spoor in zijn geheel, en hier zal ook alles van pad berekening gebeuren. Hierin zijn lijsten terug te vinden die het werkelijke spoor voorstellen in lijst vorm. Je hebt de railway-conenctions, wat alle letterlijke connecties zijn tussen alle sporen, de detection-connection, wat alleen maar detectieblokken zijn die “aan elkaar hangen” (niet letterlijk), de barrier-locations, waar alle barrières samen met hun positie opgelijst staan en de switch-connections, waar alle combinaties van connecties tussen switches opgelijst staan. In dit ADT zal er ook aan path calculation gedaan worden (**calculate-path** procedure). De path calculations is gebaseerd op het bekende DFT algoritme. Deze procedure gaat meerdere gevonden paden terug krijgen, en hierin de kleinste tussen uitkiezen, zodat je het korstte pad hebt (niet in afstand, maar in spoor bezoeken).

Naam	Signatuur	Beschrijving
make-railway	**(Ø -> Railway)**	**Constructor**
calculate-path	(Symbol, Symbolm, List -> Ø)	Dit gaat het kortste pad berekenen van de meegegeven start tot einde.

2) Infrabel

a. Infrabel

Dit is het overkoepelende adt dat ervoor zorgt dat elk component samen kunnen werken. Dit zorgt er ook voor dat het interface kan gebruikt worden. De interface zal hier gestart worden. Hierin zullen ook de procedure aanwezig zijn voor het toevoegen van de ADT’s op het interface. Hierin zijn ook alle instanties van ADT’s terug te vinden.

Naam	Signatuur	Beschrijving
make-infrabel	**(Ø -> Infrabel)**	**Constructor**
add-train!	(Symbol -> Ø)	Zal een trein toevoegen aan de lijst van treinen (hierbij wordt ook een trein object aangemaakt).
add-switch!	(Symbol -> Ø)	Zal een switch toevoegen aan de lijst van switches (hierbij wordt ook een switch object aangemaakt).

add-barrier!	(Symbol -> Ø)	Zal een barrier toevoegen aan de lijst van barriers (hierbij wordt ook een barrier object aangemaakt).
add-light!	(Symbol -> Ø)	Zal een light toevoegen aan de lijst van lichten (hierbij wordt ook een licht object aangemaakt).
add-detectionblock!	(Symbol -> Ø)	Zal een detectieblok toevoegen aan de lijst van detectieblokken (hierbij wordt ook een detectieblok object aangemaakt).
remove-train!	(Symbol -> Ø)	Zal via een train verwijderen van het interface.
start	(Ø -> Ø)	Zal de berichten sturen naar NMBS voor de detectieblokken en de switches toe te voegen.

b. Train

Dit is alles wat te maken heeft met de status van de treinen op het spoor. Hiermee kun je de snelheid van de treinen en de richtingen veranderen door het door te geven aan Z21. Dit is ook wat er meteen zal gebeuren als deze ADT aangemaakt wordt, alle waardes die het interface nodig heeft zullen rechtstreeks doorgegeven worden.

Naam	Signatuur	Beschrijving
make-infrabel-train	**(Symbol (id) -> Train)**	**Constructor**
set-speed!	(Number -> Number)	Zal via communiceren met het interface voor de snelheid van die trein aan te passen.
speed	(Ø -> Number)	Geeft de snelheid terug van de trein.
get-id	(Ø -> Symbol)	Geeft het id van de trein terug.

c. Detectieblok

Dit is alles wat te maken heeft met de detectieblokken, zoals het opvragen van de laatst gearriveerde trein. Dit ADT is eigenlijk redundant wegens dit niet meer dan een ID houder functioneert.

Naam	Signatuur	Beschrijving
make-infrabel-detectionblock	**(Symbol (id) -> Detectionblock)**	**Constructor**
get-id	(Ø -> Symbol)	Geeft het id terug van de trein.

d. Slagboom

Dit is alles wat te maken heeft met de slagbomen op het spoor en het aansturen ervan. Hier zal naar het interface gestuurd kunnen worden of deze barrière open of dicht moet,

Naam	Signatuur	Beschrijving
make-infrabel-barrier	**(Symbol (id) -> Slagboom)**	**Constructor**

set-status!	(Boolean -> Ø)	Zal als #t wordt meegegeven de slagboom open doen en als #f wordt meegegeven de slagboom sluiten.

e. Verkeerslicht

Dit is alles wat te maken heeft met de informatie over een verkeerslicht aanwezig op het spoor en het aansturen ervan met het interface.

Naam	Signatuur	Beschrijving
make-infrabel-light	**(Symbol (id) -> Verkeerslicht)**	**Constructor**
set-code!	(Symbol -> Ø)	Zal de waarde meegegeven aan het licht om zijn status aan te passen.

f. Switch

Dit is alles wat te maken heeft met de switches aanwezig op het spoor en het veranderen ervan via het interface.

Naam	Signatuur	Beschrijving
make-infrabel-switch	**(Symbol (id) -> Switch)**	**Constructor**
change-status!	(Number -> Ø)	Veranderd de stand van de switch.

g. Connection

Dit is het connectie adt dat voor het updaten van Infrabel zal zorgen door de berichten voor Infrabel in te lezen en af te handelen. Dit ADT zal zich gedragen als server. Op het moment dat deze aangemaakt wordt, zal er gewacht en geluisterd worden op inkomende connecties (bijvoorbeeld de NMBS connectie). Je kunt ook via dit ADT berichten sturen naar de clients die verbonden zijn met deze server.

Naam	Signatuur	Beschrijving
make-infrabel-connection	**(Symbol (id) -> Connection)**	**Constructor**
update-infrabel	(Ø -> Ø)	Zal de berichten voor Infrabel afhandelen naar de Infrabel componenten.
listen-for-clients	(Ø -> Ø)	Luistert naar inkomende connectie requests en zal deze af handelen.
send-message	List -> Ø	Zal het meegegeven bericht doorsturen naar de client (wat in dit geval de connectie van NMBS is).
nmbs-connection-in	Port	Geeft de input port van de NMBS connectie terug.